On Winning Conditions of High Borel Complexity in Pushdown Games

Olivier Finkel

Fundamenta Informaticae (2005) 1–22
IOS Press

On Winning Conditions of High Borel Complexity in Pushdown Games

Olivier Finkel

Equipe de Logique Mathématique

U.F.R. de Mathématiques, Université Paris 7

2 Place Jussieu 75251 Paris cedex 05, France.

finkel@logique.jussieu.fr

Abstract. In a recent paper [19, 20] Serre has presented some decidable winning conditions $\Omega_{\mathcal{A}_1 \triangleright \ldots \triangleright \mathcal{A}_n \triangleright \mathcal{A}_{n+1}}$ of arbitrarily high finite Borel complexity for games on finite graphs or on pushdown graphs.
We answer in this paper several questions which were raised by Serre in [19, 20].
We study classes $\mathbb{C}_n(A)$, defined in [20], and show that these classes are included in the class of non-ambiguous context free ω-languages. Moreover from the study of a larger class $\mathbb{C}_n^\lambda(A)$ we infer that the complements of languages in $\mathbb{C}_n(A)$ are also non-ambiguous context free ω-languages. We conclude the study of classes $\mathbb{C}_n(A)$ by showing that they are neither closed under union nor under intersection.
We prove also that there exists pushdown games, equipped with winning conditions in the form $\Omega_{\mathcal{A}_1 \triangleright \mathcal{A}_2}$, where the winning sets are not deterministic context free languages, giving examples of winning sets which are non-deterministic non-ambiguous context free languages, inherently ambiguous context free languages, or even non context free languages.

Keywords: Pushdown automata; infinite two-player games; pushdown games; winning conditions; Borel complexity; context free ω-languages; closure under boolean operations; set of winning positions.

1. Introduction

Two-player infinite games have been much studied in set theory and in particular in Descriptive Set Theory. Martin's Theorem states that every Gale Stewart game $G(A)$, where A is a Borel set, is determined, i.e. that one of the two players has a winning strategy [14].

Address for correspondence: E Mail: finkel@logique.jussieu.fr

In Computer Science, the conditions of a Gale Stewart game may be seen as a specification of a reactive system, where the two players are respectively a non terminating reactive program and the "environment". Then the problem of the synthesis of winning strategies is of great practical interest for the problem of program synthesis in reactive systems. Büchi-Landweber Theorem states that in a Gale Stewart game $G(A)$, where A is a regular ω-language, one can decide who is the winner and compute a winning strategy given by a finite state transducer.
In [23, 16] Thomas asked for an extension of this result to games played on pushdown graphs. Walukiewicz firstly showed in [25] that one can effectively construct winning strategies in parity games played on pushdown graphs and that these strategies can be computed by pushdown transducers.
Several authors have then studied pushdown games equipped with other decidable winning conditions, [4, 5, 17, 11]. Cachat, Duparc and Thomas have presented the first decidable winning condition at the Σ_3 level of the Borel hierarchy [6]. Bouquet, Serre and Walukiewicz have studied winning conditions which are boolean combinations of a Büchi condition and of the unboundedness condition which requires the stack to be unbounded, [3].
Recently Serre has given a family of decidable winning conditions of arbitrarily high finite Borel rank [19, 20]. A game between two players Adam and Eve on a pushdown graph, is equipped with a winning condition in the form $\Omega_{\mathcal{A}_1 \rhd \ldots \rhd \mathcal{A}_n \rhd \mathcal{A}_{n+1}}$, where $\mathcal{A}_1, \ldots, \mathcal{A}_n$ are deterministic pushdown automata, the stack alphabet of $\mathcal{A}_i$ being the input alphabet of $\mathcal{A}_{i+1}$, and $\mathcal{A}_{n+1}$ is a deterministic pushdown automaton with a Büchi or a parity acceptance condition. Then an infinite play is won by Eve iff during this play the stack is *strictly unbounded*, that is converges to an infinite word x and its limit $x \in L(\mathcal{A}_1 \rhd \ldots \rhd \mathcal{A}_n \rhd \mathcal{A}_{n+1})$, where $L(\mathcal{A}_1 \rhd \ldots \rhd \mathcal{A}_n \rhd \mathcal{A}_{n+1})$ is an ω-language defined as follows. A word α_0 is in $L(\mathcal{A}_1 \rhd \ldots \rhd \mathcal{A}_n \rhd \mathcal{A}_{n+1})$ iff: for all $1 \leq i \leq n$, when $\mathcal{A}_i$ reads α_{i-1} its stack is *strictly unbounded* and the limit of the stack contents is an ω-word α_i; and $\mathcal{A}_{n+1}$ accepts α_n. Serre proved that for these winning conditions one can decide the winner in a pushdown game and that the winning strategies are effective.
We solve in this paper several questions which are raised in [19, 20]. We first study the classes $\mathbb{C}_n(A)$ which contain languages in the form $L(\mathcal{A}_1 \rhd \ldots \rhd \mathcal{A}_n \rhd \mathcal{A}_{n+1})$, where A is the input alphabet of $\mathcal{A}_1$. We show that these classes are included in the class of non-ambiguous context free ω-languages. Moreover from the study of a larger class $\mathbb{C}_n^{\lambda}(A)$ we infer that the complements of languages in $\mathbb{C}_n(A)$ are also non-ambiguous context free ω-languages. We conclude the study of classes $\mathbb{C}_n(A)$ by showing that they are neither closed under union nor under intersection.
For all previously studied decidable winning conditions for pushdown games the set of winning positions for any player had been shown to be regular. In [19, 20] Serre proved that every deterministic context free language may occur as a winning set for Eve in a pushdown game equipped with a winning condition in the form $\Omega_{\mathcal{B}}$, where $\mathcal{B}$ is a deterministic pushdown automaton. The exact nature of these sets remains open and the question is raised in [19, 20] whether there exists a pushdown game equipped with a winning condition in the form $\Omega_{\mathcal{A}_1 \rhd \ldots \rhd \mathcal{A}_n \rhd \mathcal{A}_{n+1}}$ such that the set of winning positions for Eve is not a deterministic context free language. We give a positive answer to this question, giving examples of winning sets which are non-deterministic non-ambiguous context free languages, or inherently ambiguous context free languages, or even non context free languages.
The paper is organized as follows. In section 2 we recall definitions and results about pushdown automata, context free (ω)-languages, pushdown games, and winning conditions in the form $\Omega_{\mathcal{A}_1 \rhd \ldots \rhd \mathcal{A}_n \rhd \mathcal{A}_{n+1}}$. In section 3 are studied the classes $\mathbb{C}_n(A)$. Results on sets of winning positions are presented in Section 4.

2. Recall of previous definitions and results

2.1. Pushdown automata

We assume the reader to be familiar with the theory of formal (ω)-languages [22, 21, 12]. We shall use usual notations of formal language theory.

When A is a finite alphabet, a *non-empty finite word* over A is any sequence $x = a_1 \ldots a_k$, where $a_i \in A$ for $i = 1, \ldots, k$, and k is an integer ≥ 1. The *length* of x is k, denoted by $|x|$. The *empty word* has no letter and is denoted by λ; its length is 0. For $x = a_1 \ldots a_k$, we write $x(i) = a_i$ and $x[i] = x(1) \ldots x(i)$ for $i \leq k$ and $x[0] = \lambda$. $A^\star$ is the *set of finite words* (including the empty word) over A and $A^+ = A^\star - \{\lambda\}$.

The *first infinite ordinal* is ω. An ω-*word* over A is an ω-sequence $a_1 \ldots a_n \ldots$, where for all integers $i \geq 1$, $a_i \in A$. When σ is an ω-word over A, we write $\sigma = \sigma(1)\sigma(2) \ldots \sigma(n) \ldots$, where for all i, $\sigma(i) \in A$, and $\sigma[n] = \sigma(1)\sigma(2) \ldots \sigma(n)$ for all $n \geq 1$ and $\sigma[0] = \lambda$.

The *prefix relation* is denoted $\sqsubseteq$: a finite word u is a *prefix* of a finite word v (respectively, an infinite word v), denoted $u \sqsubseteq v$, if and only if there exists a finite word w (respectively, an infinite word w), such that $v = u.w$. The *set of* ω-*words* over the alphabet A is denoted by A^ω. An ω-*language* over an alphabet A is a subset of A^ω.

In [19, 20] deterministic pushdown automata are defined with two restrictions. It is supposed that there are no λ-transitions, i.e. the automata are *real time*. Moreover one can push at most one symbol in the pushdown stack using a single transition of the automaton.
We now define pushdown automata, keeping this second restriction but allowing the existence of λ-transitions; and we define also the non deterministic version of pushdown automata.

A *pushdown automaton* (PDA) is a 6-tuple $\mathcal{A} = (Q, \Gamma, A, \perp, q_{in}, \delta)$, where Q is a finite set of states, Γ is a finite pushdown alphabet, A is a finite input alphabet, $\perp$ is the bottom of stack symbol, $q_{in} \in Q$ is the initial state, and δ is the transition relation which is a mapping from $Q \times (A \cup \{\lambda\}) \times \Gamma$ to subsets of

$$\{skip(q), pop(q), push(q, \gamma) \mid q \in Q, \gamma \in \Gamma - \{\perp\}\}$$

The bottom symbol appears only at the bottom of the stack and is never popped thus for all $q, q' \in Q$ and $a \in A$, it holds that $pop(q') \notin \delta(q, a, \perp)$.

The pushdown automaton $\mathcal{A}$ is *deterministic* if for all $q \in Q$, $a \in A$ and $Z \in \Gamma$, the set $\delta(q, a, Z)$ contains at most one element; moreover if for some $q \in Q$ and $Z \in \Gamma$, $\delta(q, \lambda, Z)$ is non-empty then for all $a \in A$ the set $\delta(q, a, Z)$ is empty.

If $\sigma \in \Gamma^+$ describes the pushdown store content, the *rightmost symbol* will be assumed to be *on "top" of the store*. A configuration of the pushdown automaton $\mathcal{A}$ is a pair (q, σ) where $q \in Q$ and $\sigma \in \Gamma^\star$.
For $a \in A \cup \{\lambda\}$, $\sigma \in \Gamma^\star$ and $Z \in \Gamma$:
if $(skip(q'))$ is in $\delta(q, a, Z)$, then we write $a : (q, \sigma.Z) \mapsto_{\mathcal{A}} (q', \sigma.Z)$;

if $(pop(q'))$ is in $\delta(q, a, Z)$, then we write $a : (q, \sigma.Z) \mapsto_{\mathcal{A}} (q', \sigma)$;
if $(push(q', \gamma))$ is in $\delta(q, a, Z)$, then we write $a : (q, \sigma.Z) \mapsto_{\mathcal{A}} (q', \sigma.Z.\gamma)$.

$\mapsto^{\star}_{\mathcal{A}}$ is the transitive and reflexive closure of $\mapsto_{\mathcal{A}}$. (The subscript $\mathcal{A}$ will be omitted whenever the meaning remains clear).

Let $x = a_1 a_2 \ldots a_n$ be a finite word over A. A finite sequence of configurations $r = (q_i, \gamma_i)_{1 \leq i \leq p}$ is called a run of $\mathcal{A}$ on x, starting in configuration (q, γ), iff:

1. $(q_1, \gamma_1) = (q, \gamma)$
2. for each i, $1 \leq i \leq (p-1)$, there exists $b_i \in A \cup \{\lambda\}$ satisfying $b_i : (q_i, \gamma_i) \mapsto_{\mathcal{A}} (q_{i+1}, \gamma_{i+1})$
3. $a_1 a_2 \ldots a_n = b_1 b_2 \ldots b_{p-1}$

A run r of $\mathcal{A}$ on x, starting in configuration $(q_{in}, \perp)$, will be simply called "a run of $\mathcal{A}$ on x".

Let $x = a_1 a_2 \ldots a_n \ldots$ be an ω-word over A. An infinite sequence of configurations $r = (q_i, \gamma_i)_{i \geq 1}$ is called a run of $\mathcal{A}$ on x, starting in configuration (q, γ), iff:

1. $(q_1, \gamma_1) = (q, \gamma)$
2. for each $i \geq 1$, there exists $b_i \in A \cup \{\lambda\}$ satisfying $b_i : (q_i, \gamma_i) \mapsto_{\mathcal{A}} (q_{i+1}, \gamma_{i+1})$
3. either $a_1 a_2 \ldots a_n \ldots = b_1 b_2 \ldots b_n \ldots$
   or $b_1 b_2 \ldots b_n \ldots$ is a finite prefix of $a_1 a_2 \ldots a_n \ldots$

The run r is said to be complete when $a_1 a_2 \ldots a_n \ldots = b_1 b_2 \ldots b_n \ldots$
A complete run r of $\mathcal{A}$ on x, starting in configuration $(q_{in}, \perp)$, will be simply called "a run of $\mathcal{A}$ on x".

If the pushdown automaton $\mathcal{A}$ is equipped with a set of final states $F \subseteq Q$,
the finitary language *accepted by* $(\mathcal{A}, F)$ is :

$$L^f(\mathcal{A}, F) = \{x \in A^\star \mid \text{ there exists a run } r = (q_i, \gamma_i)_{1 \leq i \leq p} \text{ of } \mathcal{A} \text{ on } x \text{ such that } q_p \in F\}$$

The class CFL of *context free languages* is the class of finitary languages which are accepted by pushdown automata by final states.
Notice that other accepting conditions by PDA have been shown to be equivalent to the acceptance condition by final states. Let us cite, [1]: (a) acceptance by empty storage, (b) acceptance by final states and empty storage, (c) acceptance by topmost stack letter, (d) acceptance by final states and topmost stack letter.
The class $DCFL$ of *deterministic context free languages* is the class of finitary languages which are accepted by deterministic pushdown automata (DPDA) by final states.
Notice that for DPDA, acceptance by final states is not equivalent to acceptance by empty storage: this is due to the fact that a language accepted by a DPDA by empty storage must be *prefix-free* while this is not necessary in the case of acceptance by final states [1].

The ω-language *Büchi accepted by* $(\mathcal{A}, F)$ is :

$$L(\mathcal{A}, F) = \{x \in A^\omega \mid \text{ there exists a run } r \text{ of } \mathcal{A} \text{ on } x \text{ such that } In(r) \cap F \neq \emptyset\}$$

where $In(r)$ is the set of all states entered infinitely often during run r.

If instead the pushdown automaton $\mathcal{A}$ is equipped with a set of accepting sets of states $\mathcal{F} \subseteq 2^Q$, the ω-language *Muller accepted by* $(\mathcal{A}, \mathcal{F})$ is :

$$L(\mathcal{A}, \mathcal{F}) = \{x \in A^\omega \mid \text{ there exists a run } r \text{ of } \mathcal{A} \text{ on } x \text{ such that } In(r) \in \mathcal{F}\}$$

The class CFL_ω of *context free ω-languages* is the class of ω-languages which are Büchi or Muller accepted by pushdown automata.

Another usual acceptance condition for ω-words is the parity condition. In that case a pushdown automaton $\mathcal{A} = (Q, \Gamma, A, \perp, q_{in}, \delta)$ is equipped with a function col from Q to a finite set of colors $C \subset \mathbb{N}$. The ω-language accepted by $(\mathcal{A}, col)$ is:

$$L(\mathcal{A}, col) = \{x \in A^\omega \mid \text{ there exists a run } r \text{ of } \mathcal{A} \text{ on } x \text{ such that } sc(r) \text{ is even }\}$$

where $sc(r)$ is the smallest color appearing infinitely often in the run r.
It is easy to see that a Büchi acceptance condition can be expressed as a parity acceptance condition which itself can be expressed as a Muller condition.
Thus the class of ω-languages which are accepted by pushdown automata with a parity acceptance condition is still the class CFL_ω.

Consider now *deterministic* pushdown automata. If $\mathcal{A}$ is a deterministic pushdown automaton, then for every $\sigma \in A^\omega$, there exists at most one run r of $\mathcal{A}$ on σ determined by the starting configuration. The pushdown automaton has the continuity property iff for every $\sigma \in A^\omega$, there exists a unique run of $\mathcal{A}$ on σ and this run is complete. It is shown in [8] that each ω-language accepted by a deterministic Büchi (respectively, Muller) pushdown automaton can be accepted by a deterministic Büchi (respectively, Muller) pushdown automaton with the continuity property. The same proof works in the case of deterministic pushdown automata with parity acceptance condition.

The class of ω-languages accepted by deterministic Büchi pushdown automata is a strict subclass of the class $DCFL_\omega$ of ω-languages accepted by deterministic pushdown automata with a Muller condition.
One can easily show that $DCFL_\omega$ is also the class of ω-languages accepted by DPDA with a parity acceptance condition.
Each ω -language in $DCFL_\omega$ can be accepted by a deterministic pushdown automaton *having the continuity property* with parity (or Muller) acceptance condition. One can then show that the class $DCFL_\omega$ is closed under complementation.

The notion of ambiguity for context free ω-languages has been firstly studied in [10]. A context free ω-language is non ambiguous iff it is accepted by a Büchi or Muller pushdown automaton such that every ω-word on the input alphabet has at most one accepting run. Notice that we consider here that two runs

are equal iff they go through the same infinite sequence of configurations *and* λ-transitions occur at the same steps of the computations.
The class $NA - CFL_\omega$ is the class of non ambiguous context free ω-languages.
The inclusion $DCFL_\omega \subseteq NA - CFL_\omega$ will be useful in the sequel. We shall denote $Co - NA - CFL_\omega$ the class of complements of non ambiguous context free ω-languages.

2.2. Pushdown games

Recall first that a *pushdown process* may be viewed as a PDA without input alphabet and initial state. A pushdown process is a 4-tuple $\mathcal{P} = (Q, \Gamma, \perp, \delta)$, where Q is a finite set of states, Γ is a finite pushdown alphabet, $\perp$ is the bottom of stack symbol, and δ is the transition relation which is a mapping from $Q \times \Gamma$ to subsets of

$$\{skip(q), pop(q), push(q, \gamma) \mid q \in Q, \gamma \in \Gamma - \{\perp\}\}$$

Configurations of a pushdown process are defined as for PDA. A configuration of the pushdown process $\mathcal{P}$ is a pair (q, σ) where $q \in Q$ and $\sigma \in \Gamma^\star$.

To a pushdown process $\mathcal{P} = (Q, \Gamma, \perp, \delta)$ is naturally associated a pushdown graph $G = (V, \rightarrow)$ which is a directed graph. The set of vertices V is the set of configurations of $\mathcal{P}$. The edge relation $\rightarrow$ is defined as follows: $(q, \sigma) \rightarrow (q', \sigma')$ iff the configuration (q', σ') can be reached in one transition of $\mathcal{P}$ from the configuration (q, σ).

We shall consider in the sequel infinite games between two players named Eve and Adam on such pushdown graphs.
So we shall assume that the set Q of states of a pushdown process is partitioned in two sets Q_E and Q_A. A configuration (q, σ) is in V_E iff q is in Q_E and it is in V_A iff q is in Q_A so (V_E, V_A) is a partition of the set of configurations V.

The game graph $(V_E, V_A, \rightarrow)$ is called a *pushdown game graph*.
A play from a vertex v_1 of this graph is defined as follows. If $v_1 \in V_E$, Eve chooses a vertex v_2 such that $v_1 \rightarrow v_2$; otherwise Adam chooses such a vertex. If there is no such vertex v_2 the play stops. Otherwise the play may continue. If $v_2 \in V_E$, Eve chooses a vertex v_3 such that $v_2 \rightarrow v_3$; otherwise Adam chooses such a vertex. If there is no such vertex v_3 the play stops. Otherwise the play continues in the same way. So a play starting from the vertex v_1 is a *finite or infinite* sequence of vertices $v_1 v_2 v_3 \ldots$ such that for all i $v_i \rightarrow v_{i+1}$. We may assume, as in [19, 20], that in fact all plays are infinite.

A *winning condition* for Eve is a set $\Omega \subseteq V^\omega$. An infinite two-player pushdown game is a 4-tuple $(V_E, V_A, \rightarrow, \Omega)$, where $(V_E, V_A, \rightarrow)$ is a pushdown game graph and $\Omega \subseteq V^\omega$ is a winning condition for Eve.
In a pushdown game equipped with the winning condition Ω, Eve wins a play $v_1 v_2 v_3 \ldots$ iff $v_1 v_2 v_3 \ldots \in \Omega$.
A *strategy* for Eve is a partial function $f : V^\star.V_E \rightarrow V$ such that, for all $x \in V^\star$ and $v \in V_E$, $v \rightarrow f(x.v)$.
Eve uses the strategy f in a play $v_1 v_2 v_3 \ldots$ iff for all $v_i \in V_E$, $v_{i+1} = f(v_1 v_2 \ldots v_i)$.

A strategy f is a *winning strategy* for Eve from some position v_1 iff Eve wins all plays starting from v_1 and during which she uses the strategy f.
A vertex $v \in V$ is a *winning position* for Eve iff she has a winning strategy from it.
The notions of winning strategy and winning position are defined for the other player Adam in a similar way.
The set of winning positions for Eve and Adam will be respectively denoted by W_E and W_A.

2.3. Winning condition $\Omega_{\mathcal{A}_1 \triangleright \ldots \triangleright \mathcal{A}_n \triangleright \mathcal{A}_{n+1}}$

We first recall the definition of ω-languages in the form $L(\mathcal{A}_1 \triangleright \ldots \triangleright \mathcal{A}_n \triangleright \mathcal{A}_{n+1})$ which are used in [19, 20] to define the winning conditions $\Omega_{\mathcal{A}_1 \triangleright \ldots \triangleright \mathcal{A}_n \triangleright \mathcal{A}_{n+1}}$.

We shall need the notion of limit of an infinite sequence of finite words over some finite alphabet A.
Let then $(\beta_n)_{n \geq 0}$ be an infinite sequence of words $\beta_n \in A^\star$. The finite *or* infinite word $\lim_{n \in \omega} \beta_n$ is determined by the set of its (finite) prefixes: for all v in $A^\star$,
$v \sqsubseteq \lim_{n \in \omega} \beta_n \leftrightarrow \exists n \forall p \geq n \quad \beta_p[|v|] = v$.

Let now $\mathcal{A} = (Q, \Gamma, A, \perp, q_{in}, \delta)$ be a pushdown automaton reading words over the alphabet A and let $\alpha \in A^\omega$. The pushdown stack of $\mathcal{A}$ is said to be *strictly unbounded* during a run $r = (q_i, \gamma_i)_{i \geq 1}$ of $\mathcal{A}$ on α iff $\lim_{n \geq 1} \gamma_n$ is infinite.

We define now ω-languages $L(\mathcal{A}_1 \triangleright \ldots \triangleright \mathcal{A}_n \triangleright \mathcal{A}_{n+1})$ in a slightly more general case than in [20], because this will be useful in the next section. Notice that in [20], these ω-languages are only defined in the case where $\mathcal{A}_1, \ldots, \mathcal{A}_n$, are *real-time* deterministic pushdown automata, and $\mathcal{A}_{n+1}$ is a *real-time* deterministic pushdown automaton equipped with a parity or a Büchi acceptance condition.

Let n be an integer ≥ 0 and $\mathcal{A}_1$, $\mathcal{A}_2$, $\ldots \mathcal{A}_n$, be some deterministic pushdown automata (in the case $n = 0$ there are not any such automata).
Let $(\mathcal{A}_{n+1}, \mathcal{C})$ be a pushdown automaton equipped with a Büchi or a parity acceptance condition.
The input alphabet of $\mathcal{A}_1$ is denoted A and we assume that, for each integer $i \in [1, n]$, the input alphabet of $\mathcal{A}_{i+1}$ is the stack alphabet of $\mathcal{A}_i$.

We define inductively the ω-language $L(\mathcal{A}_1 \triangleright \ldots \triangleright \mathcal{A}_n \triangleright \mathcal{A}_{n+1}) \subseteq A^\omega$ by:

1. If $n = 0$, $L(\mathcal{A}_1 \triangleright \ldots \triangleright \mathcal{A}_n \triangleright \mathcal{A}_{n+1}) = L(\mathcal{A}_{n+1}, \mathcal{C})$ is the ω-language accepted by $\mathcal{A}_{n+1}$ with acceptance condition $\mathcal{C}$.
2. If $n > 0$, $L(\mathcal{A}_1 \triangleright \ldots \triangleright \mathcal{A}_n \triangleright \mathcal{A}_{n+1})$ is the set of ω-words $\alpha \in A^\omega$ such that:
   - When $\mathcal{A}_1$ reads α, the stack of $\mathcal{A}_1$ is strictly unbounded hence the sequence of stack contents has an infinite limit α_1.
   - $\alpha_1 \in L(\mathcal{A}_2 \triangleright \ldots \triangleright \mathcal{A}_n \triangleright \mathcal{A}_{n+1})$.

Let now $(V_E, V_A, \rightarrow)$ be a pushdown game graph associated with a pushdown process $\mathcal{P}$. An infinite play $v_1 v_2 v_3 \ldots$, where $v_i = (q_i, \gamma_i)$, is in the set $\Omega_{\mathcal{A}_1 \triangleright \ldots \triangleright \mathcal{A}_n \triangleright \mathcal{A}_{n+1}}$ iff:

1. The pushdown stack of $\mathcal{P}$ is *strictly unbounded* during the play, i.e. $\lim_{n\geq 1}\gamma_n$ is infinite, and
2. $\lim_{n\geq 1}\gamma_n \in L(\mathcal{A}_1 \triangleright \ldots \triangleright \mathcal{A}_n \triangleright \mathcal{A}_{n+1})$.

3. Classes $\mathbb{C}_n(A)$

3.1. Classes $\mathbb{C}_n(A)$ and context free ω-languages

For each integer $n \geq 0$ and each finite alphabet A the class $\mathbb{C}_n(A)$ is defined in [20] as the class of ω-languages in the form $L(\mathcal{A}_1 \triangleright \ldots \triangleright \mathcal{A}_n \triangleright \mathcal{A}_{n+1})$, where $\mathcal{A}_1, \ldots, \mathcal{A}_n$, are *real-time* deterministic pushdown automata, the input alphabet of $\mathcal{A}_1$ being A, and $\mathcal{A}_{n+1}$ is a *real-time* deterministic pushdown automaton equipped with a parity acceptance condition. It is easy to see that we obtain the same class $\mathbb{C}_n(A)$ if we restrict the definition to the case of *real-time* deterministic pushdown automata $\mathcal{A}_1, \ldots, \mathcal{A}_n, \mathcal{A}_{n+1}$, having the *continuity property*.
We shall denote $\mathbb{C}_n^\lambda(A)$ the class obtained in the same way except that the deterministic pushdown automata $\mathcal{A}_1, \ldots, \mathcal{A}_n, \mathcal{A}_{n+1}$, having still the continuity property, may have λ-transitions, i.e. may be non real time.

In the sequel of this paper when we consider languages in the form $L(\mathcal{A}_1 \triangleright \ldots \triangleright \mathcal{A}_n \triangleright \mathcal{A}_{n+1})$, we shall always implicitely assume that the pushdown automata $\mathcal{A}_1, \ldots, \mathcal{A}_n, \mathcal{A}_{n+1}$, have the *continuity property*, and that, for each integer $i \in [1, n]$, the input alphabet of $\mathcal{A}_{i+1}$ is the stack alphabet of $\mathcal{A}_i$.

In order to prove that classes $\mathbb{C}_n(A)$, $\mathbb{C}_n^\lambda(A)$, are included in the class of context free ω-languages we first state the following lemma.

Lemma 3.1. Let $\mathcal{A}_1 = (Q_1, \Gamma_1, A_1, \perp_1, q_0^1, \delta_1)$ be a deterministic pushdown automaton and $\mathcal{A}_2 = (Q_2, \Gamma_2, \Gamma_1, \perp_2, q_0^2, \delta_2)$ be a pushdown automaton equipped with a set of final states $F \subseteq Q_2$. Then the ω-language $L(\mathcal{A}_1 \triangleright \mathcal{A}_2)$ is a context free ω-language.

Proof. Let $\mathcal{A}_1 = (Q_1, \Gamma_1, A_1, \perp_1, q_0^1, \delta_1)$ be a deterministic pushdown automaton and $\mathcal{A}_2 = (Q_2, \Gamma_2, \Gamma_1, \perp_2, q_0^2, \delta_2)$ be a pushdown automaton equipped with a set of final states $F \subseteq Q_2$.

Recall that an ω-word $\alpha \in A_1^\omega$ is in $L(\mathcal{A}_1 \triangleright \mathcal{A}_2)$ iff:

- When $\mathcal{A}_1$ reads α, the stack of $\mathcal{A}_1$ is strictly unbounded hence the sequence of stack contents has an infinite limit α_1.
- $\alpha_1 \in L(\mathcal{A}_2, F)$.

We can decompose the reading of an ω-word $\alpha \in L(\mathcal{A}_1 \triangleright \mathcal{A}_2)$ by the pushdown automaton $\mathcal{A}_1$ in the following way.
When reading α, $\mathcal{A}_1$ goes through the infinite sequence of configurations $(q_i, \gamma_i)_{i\geq 1}$. The infinite sequence of stack contents $(\gamma_i)_{i\geq 1}$ has limit α_1 thus for each integer $j \geq 1$, there is a smallest integer n_j such that, for all integers $i \geq n_j$, $\alpha_1[j] = \gamma_i[j]$.

The word α can then be decomposed in the form

$$\alpha = \sigma_1.\sigma_2 \ldots \sigma_n \ldots$$

where for all integers $j \geq 1$, $\sigma_j \in A_1^\star$ and

$$\sigma_j : (q_{n_j}, \alpha_1[j]) \mapsto^\star_{\mathcal{A}_1} (q_{n_{j+1}}, \alpha_1[j+1]) = (q_{n_{j+1}}, \alpha_1[j].\alpha_1(j+1))$$

Notice that $n_1 = 1$, $q_1 = q_0^1$ and $\alpha_1[1] = \perp_1$ hence $\sigma_1 : (q_0^1, \perp_1) \mapsto^\star_{\mathcal{A}_1} (q_{n_2}, \alpha_1[2])$.

Let now, for each $q, q' \in Q_1$ and $a, b \in \Gamma_1$, the language $\mathcal{L}_{(q,q',a,b)}$ be the set of words $\sigma \in A_1^\star$ such that: $\sigma : (q, a) \mapsto^\star_{\mathcal{A}_1} (q', a.b)$. This language of finite words over A_1 is accepted by the pushdown automaton $\mathcal{A}_1$ with the following modifications: the initial configuration is (q, a) and the acceptance is by final state q' *and* by final stack content $a.b$. It is easy to see that this language is also accepted by a deterministic pushdown automaton by final states so it is in the class $DCFL$.

Then each word σ_j belongs to the deterministic context free language

$$\mathcal{L}_{(q_{n_j}, q_{n_{j+1}}, \alpha_1(j), \alpha_1(j+1))} = \{\sigma \in A_1^\star \mid \sigma : ((q_{n_j}, \alpha_1(j)) \mapsto^\star_{\mathcal{A}_1} (q_{n_{j+1}}, \alpha_1(j).\alpha_1(j+1))\}$$

In order to describe the ω-language $L(\mathcal{A}_1 \rhd \mathcal{A}_2)$ from the ω-language $L(\mathcal{A}_2, F)$ and the deterministic context free languages $\mathcal{L}_{(q,q',a,b)}$, for $q, q' \in Q_1$ and $a, b \in \Gamma_1$, we now recall the notion of substitution.

A *substitution* is a mapping $f : \Sigma \to 2^{\Gamma^\star}$, where Σ and Γ are two finite alphabets. If $\Sigma = \{a_1, \ldots, a_n\}$, then for all integers $i \in [1; n]$, $f(a_i) = L_i$ is a finitary language over the alphabet Γ.
Now this mapping is extended in the usual manner to finite words: for all letters $a_{i_1}, \ldots, a_{i_n} \in \Sigma$, $f(a_{i_1} \ldots a_{i_n}) = f(a_{i_1}) \ldots f(a_{i_n})$, and to finitary languages $L \subseteq \Sigma^\star$: $f(L) = \cup_{x \in L} f(x)$.
If for each letter $a \in \Sigma$, the language $f(a)$ does not contain the empty word, then the substitution is said to be λ-free and the mapping f may be extended to ω-words:

$$f(x(1) \ldots x(n) \ldots) = \{u_1 \ldots u_n \ldots \mid \forall i \geq 1 \quad u_i \in f(x(i))\}$$

and to ω-languages $L \subseteq \Sigma^\omega$ by setting $f(L) = \cup_{x \in L} f(x) \subseteq \Gamma^\omega$.
If the substitution is not λ-free we can define $f(L)$ in the same way for $L \subseteq \Sigma^\omega$ but this time $f(L) \subseteq \Gamma^\star \cup \Gamma^\omega$, i.e. $f(L)$ may contain finite *or* infinite words.
The substitution f is said to be a context free substitution iff for all $a \in \Sigma$ the finitary language $f(a)$ is context free.
Recall that Cohen and Gold proved in [7] that if L is a context free ω-language and f is a context free substitution then $f(L) \cap \Gamma^\star$ and $f(L) \cap \Gamma^\omega$ are context free.

We define now a new alphabet

$$\Delta = \{L(q, q', a, b) \mid q, q' \in Q_1 \text{ and } a, b \in \Gamma_1\}$$

and we consider the substitution $h : \Gamma_1 \to 2^\Delta$ defined, for all $b \in \Gamma_1$, by:

$$h(b) = \{L(q, q', a, b) \mid q, q' \in Q_1 \text{ and } a \in \Gamma_1\}$$

Applying this substitution to the ω-language $L(\mathcal{A}_2, F) \subseteq \Gamma_1^\omega$, we get $h(L(\mathcal{A}_2, F))$. The substitution h is λ-free thus $h(L(\mathcal{A}_2, F))$ is a ω-language over Δ. Moreover for each $b \in \Gamma_1$ the set $h(b)$ is finite hence context free. Thus $h(L(\mathcal{A}_2, F)) \subseteq \Delta^\omega$ is a context free ω-language because $L(\mathcal{A}_2, F)$ is a context free ω-language and the substitution h is a context free substitution.

Let now $R \subseteq \Delta^\omega$ be the ω-language defined as follows. An ω-word $x \in R$ has its first letter in the set $\{L(q_0^1, q', \perp_1, b) \mid q' \in Q_1 \text{ and } b \in \Gamma_1\}$, and each letter $L(q, q', a, b)$, for $q, q' \in Q_1$ and $a, b \in \Gamma_1$, in x is followed by a letter in the set $\{L(q', q'', b, c) \mid q'' \in Q_1 \text{ and } c \in \Gamma_1\}$.
The ω-language R is regular thus $h(L(\mathcal{A}_2, F)) \cap R \subseteq \Delta^\omega$ is a context free ω-language because the class CFL_ω is closed under intersection with regular ω-languages [7].

Consider now the substitution $\Theta : \Delta \rightarrow 2^{A_1^\star}$ defined, for all letters $L(q, q', a, b) \in \Delta$, by $\Theta(L(q, q', a, b)) = \mathcal{L}_{(q,q',a,b)}$. The substitution Θ is context free thus

$$\Theta[h(L(\mathcal{A}_2, F)) \cap R] \cap A_1^\omega$$

is a context free ω-language and so is $\perp_1.(\ \Theta[h(L(\mathcal{A}_2, F)) \cap R] \cap A_1^\omega\ )$. By construction this ω-language is $L(\mathcal{A}_1 \triangleright \mathcal{A}_2)$. $\square$

We can in fact obtain a refined result if the language $L(\mathcal{A}_2, F)$ is non ambiguous.

Lemma 3.2. Let $\mathcal{A}_1 = (Q_1, \Gamma_1, A_1, \perp_1, q_0^1, \delta_1)$ be a deterministic pushdown automaton and $\mathcal{A}_2 = (Q_2, \Gamma_2, \Gamma_1, \perp_2, q_0^2, \delta_2)$ be a pushdown automaton equipped with a set of final states $F \subseteq Q_2$. If the ω-language $L(\mathcal{A}_2, F)$ is non ambiguous then $L(\mathcal{A}_1 \triangleright \mathcal{A}_2) \in NA - CFL_\omega$.

Proof. Let $\mathcal{A}_1 = (Q_1, \Gamma_1, A_1, \perp_1, q_0^1, \delta_1)$ be a deterministic pushdown automaton and $\mathcal{A}_2 = (Q_2, \Gamma_2, \Gamma_1, \perp_2, q_0^2, \delta_2)$ be a pushdown automaton equipped with a set of final states $F \subseteq Q_2$.

We assume that $L(\mathcal{A}_2, F)$ is non ambiguous so we can assume, without loss of generality, that the pushdown automaton $\mathcal{A}_2$ itself is non ambiguous.

We are going to explain informally the construction of a non ambiguous Büchi pushdown automaton $\mathcal{A}$ accepting the ω-language $L(\mathcal{A}_1 \triangleright \mathcal{A}_2)$.

We refer now to the proof of the preceding lemma. We have considered the reading of an ω-word $\alpha \in L(\mathcal{A}_1 \triangleright \mathcal{A}_2)$ by $\mathcal{A}_1$, and we have shown that the word α can then be decomposed in the form

$$\alpha = \sigma_1.\sigma_2 \ldots \sigma_n \ldots$$

where for all integers $j \geq 1$, σ_j belongs to the deterministic context free language

$$\mathcal{L}_{(q_{n_j}, q_{n_{j+1}}, \alpha_1(j), \alpha_1(j+1))} = \{\sigma \in A_1^\star \mid \sigma : ((q_{n_j}, \alpha_1(j)) \mapsto^\star_{\mathcal{A}_1} (q_{n_{j+1}}, \alpha_1(j).\alpha_1(j+1))\}$$

We can see that the integers n_j were defined in a unique way. However there may exist several decompositions of the ω-word α into words of languages $\mathcal{L}_{(q,q',a,b)}$.
In order to ensure a unique decomposition we are going to slightly modify the definition of these languages.
For each $q, q' \in Q_1$ and $a, b \in \Gamma_1$, the language $\mathcal{U}_{(q,q',a,b)}$ is the set of words $\sigma \in A_1^\star$ such that:

(a) $\sigma : (q, a) \mapsto^{\star}_{\mathcal{A}_1} (q', a.b)$ and

(b) If for some $\sigma' \sqsubset \sigma$ and $s \in Q$, $\sigma' : (q, a) \mapsto^{\star}_{\mathcal{A}_1} (s, a.b)$ then there is a word $u \in A_1^{\star}$ and a state $t \in Q$, such that $\sigma'.u \sqsubseteq \sigma$ and $u : (s, a.b) \mapsto^{\star}_{\mathcal{A}_1} (t, a)$.

(c) If there is a run $(q_i, \gamma_i)_{1 \leq i \leq p}$ of $\mathcal{A}_1$ on σ such that $(q_1, \gamma_1) = (q, a)$ and $(q_p, \gamma_p) = (s, a.b)$ for some $s \in Q$, $s \neq q'$, then either there is an integer $p' < p$ such that $(q_i, \gamma_i)_{1 \leq i \leq p'}$ is a run of $\mathcal{A}_1$ on σ and $(q_{p'}, \gamma_{p'}) = (q', a.b)$ or it holds that $\lambda : (s, a.b) \mapsto^{\star}_{\mathcal{A}_1} (s', a)$ for some $s' \in Q$ and $\lambda : (s', a) \mapsto^{\star}_{\mathcal{A}_1} (q', a.b)$.

It is easy to see that the languages $\mathcal{U}_{(q,q',a,b)}$ are also in the class $DCFL$ and that, for each $q, q' \in Q_1$ and $a, b \in \Gamma_1$, it holds that $\mathcal{U}_{(q,q',a,b)} \subseteq \mathcal{L}_{(q,q',a,b)}$.
We can see that, in the above decomposition $\alpha = \sigma_1.\sigma_2 \ldots \sigma_n \ldots$ of the ω-word α, for all integers $j \geq 1$, the word σ_j belongs in fact to the deterministic context free language $\mathcal{U}_{(q_{n_j}, q_{n_{j+1}}, \alpha_1(j), \alpha_1(j+1))}$.

The rest of the proof of Lemma 3.1 can be pursued, replacing languages $\mathcal{L}_{(q,q',a,b)}$ by languages $\mathcal{U}_{(q,q',a,b)}$.

But now we have a unique decomposition of α in the form

$$\alpha = \sigma'_1.\sigma'_2 \ldots \sigma'_n \ldots$$

where for all integers $j \geq 1$, the word σ'_j belongs to some language $\mathcal{U}_{(s_j, t_j, a_j, b_j)}$ satisfying: (1) $s_1 = q_0^1$, $a_1 = \perp_1$, (2) for all integers $j \geq 1$, $t_j = s_{j+1}$ and $b_j = a_{j+1}$.

This unique decomposition is crucial in the construction of the non ambiguous Büchi PDA $\mathcal{A}$ accepting $L(\mathcal{A}_1 \rhd \mathcal{A}_2)$. We shall explain informally the behaviour of this automaton.
For each $q, q' \in Q_1$ and $a, b \in \Gamma_1$, the language $\mathcal{U}_{(q,q',a,b)}$ is accepted by a deterministic pushdown automaton $\mathcal{B}^{(q,q',a,b)}$ whose stack alphabet is denoted $\Gamma^{(q,q',a,b)}$. We can assume that all these alphabets are disjoint and that they are also disjoint from Γ_1, the stack alphabet of $\mathcal{A}_1$. The stack alphabet of $\mathcal{A}$ will be

$$\Gamma^{\mathcal{A}} = \Gamma_1 \cup \bigcup_{q,q' \in Q_1 \ and \ a,b \in \Gamma_1} \Gamma^{(q,q',a,b)}$$

When reading an ω-word $\alpha \in L(\mathcal{A}_1 \rhd \mathcal{A}_2)$ the pushdown automaton $\mathcal{A}$ will guess, using the non determinism, the **unique** decomposition of α in the form

$$\alpha = \sigma'_1.\sigma'_2 \ldots \sigma'_n \ldots$$

where for all integers $j \geq 1$, the word σ'_j belongs to some language $\mathcal{U}_{(s_j, t_j, a_j, b_j)}$ satisfying: (1) $s_1 = q_0^1$, $a_1 = \perp_1$, (2) for all integers $j \geq 1$, $t_j = s_{j+1}$ and $b_j = a_{j+1}$.
In addition $\mathcal{A}$ will simulate the reading of the ω-word $\alpha_1 = a_1 a_2 a_3 \ldots$ by the PDA $\mathcal{A}_2$.

During a run of $\mathcal{A}$ the stack content is always a word in the form $\perp.u.v$ where $\perp$ is the bottom symbol of $\mathcal{A}$, $u \in (\Gamma_1 - \{\perp\})^{\star}$ and v is in $(\Gamma^{(q,q',a,b)})^{\star}$ for some $q, q' \in Q_1$ and $a, b \in \Gamma_1$.

After having read the initial segment $\sigma'_1.\sigma'_2 \ldots \sigma'_j$ of α, the content of the stack of $\mathcal{A}$ is equal to the content of the stack of $\mathcal{A}_2$ after having read $a_1 a_2 \ldots a_j$.

Then $\mathcal{A}$ guesses that the next word in the decomposition of α belongs to some $\mathcal{U}_{(s_{j+1},t_{j+1},a_{j+1},b_{j+1})}$. It uses the stack alphabet $\Gamma^{(s_{j+1},t_{j+1},a_{j+1},b_{j+1})}$ on the top of the stack to simulate the reading of σ'_{j+1} by $\mathcal{B}^{(s_{j+1},t_{j+1},a_{j+1},b_{j+1})}$. Then when it has guessed that it has completely read the word σ'_{j+1}, it erases letters of $\Gamma^{(s_{j+1},t_{j+1},a_{j+1},b_{j+1})}$ from the stack, and simulates the reading of the letter a_{j+1} by $\mathcal{A}_2$, and so on.
A Büchi acceptance condition is then used to simulate the acceptance of α_1 by $\mathcal{A}_2$.

The Büchi PDA $(\mathcal{A}_2, F)$ is non ambiguous and the above cited decomposition of α is unique so there is a unique accepting run of the Büchi PDA $\mathcal{A}$ on α.
Finally we have proved that $L(\mathcal{A}_1 \rhd \mathcal{A}_2) \in NA - CFL_\omega$. □

Proposition 3.3. Let n be an integer ≥ 1, $\mathcal{A}_1, \mathcal{A}_2, \ldots \mathcal{A}_n$, be some deterministic pushdown automata and $(\mathcal{A}_{n+1}, \mathcal{C})$ be a pushdown automaton equipped with a Büchi acceptance condition. The input alphabet of $\mathcal{A}_1$ is denoted A and we assume that, for each integer $i \in [1, n]$, the input alphabet of $\mathcal{A}_{i+1}$ is the stack alphabet of $\mathcal{A}_i$. Then $L(\mathcal{A}_1 \rhd \ldots \rhd \mathcal{A}_n \rhd \mathcal{A}_{n+1}) \in CFL_\omega$. Moreover if $L(\mathcal{A}_{n+1}, \mathcal{C})$ is non ambiguous then $L(\mathcal{A}_1 \rhd \ldots \rhd \mathcal{A}_n \rhd \mathcal{A}_{n+1}) \in NA - CFL_\omega$.

Proof. We reason by induction on the integer n.
For $n = 1$ the result is stated in the above Lemmas 3.1 and 3.2.

Assume now that the result is true for some integer $n \geq 1$.
Let $\mathcal{A}_1$, $\mathcal{A}_2$, $\ldots \mathcal{A}_n$, $\mathcal{A}_{n+1}$, be some deterministic pushdown automata and $(\mathcal{A}_{n+2}, \mathcal{C})$ be a pushdown automaton equipped with a Büchi acceptance condition such that the language $L(\mathcal{A}_1 \rhd \ldots \rhd \mathcal{A}_{n+1} \rhd \mathcal{A}_{n+2}) \subseteq A^\omega$ is well defined.
By induction hypothesis the language $L(\mathcal{A}_2 \rhd \ldots \rhd \mathcal{A}_{n+1} \rhd \mathcal{A}_{n+2})$ is a context free ω-language accepted by a Büchi pushdown automaton $(\mathcal{A}, F)$.
But by definition of the language $L(\mathcal{A}_1 \rhd \ldots \rhd \mathcal{A}_{n+1} \rhd \mathcal{A}_{n+2})$ it holds that

$$L(\mathcal{A}_1 \rhd \ldots \rhd \mathcal{A}_{n+1} \rhd \mathcal{A}_{n+2}) = L(\mathcal{A}_1 \rhd \mathcal{A})$$

thus Lemma 3.1 implies that $L(\mathcal{A}_1 \rhd \ldots \rhd \mathcal{A}_{n+1} \rhd \mathcal{A}_{n+2}) \in CFL_\omega$.

Assume now that $L(\mathcal{A}_{n+1}, \mathcal{C})$ is non ambiguous. Reasoning as above but applying Lemma 3.2 instead of Lemma 3.1 we infer that $L(\mathcal{A}_1 \rhd \ldots \rhd \mathcal{A}_{n+1} \rhd \mathcal{A}_{n+2})$ is in $NA - CFL_\omega$. □

In particular, Proposition 3.3 implies the following result.

Corollary 3.4. For each integer $n \geq 0$, the following inclusions hold:

$$\mathbb{C}_n(A) \subseteq \mathbb{C}_n^\lambda(A) \subseteq NA - CFL_\omega$$

We shall later get a stronger result (see Corollary 3.8) from the study of closure properties of classes $\mathbb{C}_n(A)$, $\mathbb{C}_n^\lambda(A)$.

3.2. Closure properties of classes $\mathbb{C}_n(A)$, $\mathbb{C}_n^\lambda(A)$

We first state the following lemma.

Lemma 3.5. The class $\mathbb{C}_1^\lambda(A)$ is closed under complementation.

Proof. Let $\mathcal{A}_1 = (Q_1, \Gamma_1, A_1, \perp_1, q_0^1, \delta_1)$ be a deterministic pushdown automaton and $(\mathcal{A}_2 = (Q_2, \Gamma_2, \Gamma_1, \perp_2, q_0^2, \delta_2), col_2)$ be a deterministic pushdown automaton equipped with a parity acceptance condition. Recall that an ω-word $\alpha \in A_1^\omega$ is in $L(\mathcal{A}_1 \rhd \mathcal{A}_2)$ iff: when $\mathcal{A}_1$ reads α, the stack of $\mathcal{A}_1$ is strictly unbounded and the sequence of stack contents has an infinite limit $\alpha_1 \in L(\mathcal{A}_2, col_2)$.

Thus an ω-word $\alpha \in A_1^\omega$ is in the complement of $L(\mathcal{A}_1 \rhd \mathcal{A}_2)$ iff one of the two following conditions holds:

(1) When $\mathcal{A}_1$ reads α, the stack of $\mathcal{A}_1$ is strictly unbounded and the limit α_1 of stack contents is in the complement of $L(\mathcal{A}_2, col_2)$.

(2) When $\mathcal{A}_1$ reads α, the stack of $\mathcal{A}_1$ is *not strictly unbounded*.

The class $DCFL_\omega$ is closed under complementation thus the complement of $L(\mathcal{A}_2, col_2)$ is equal to $L(\mathcal{A}_3, col_3)$, for some deterministic pushdown automaton $\mathcal{A}_3$ equipped with a parity acceptance condition.
The language $L(\mathcal{A}_1 \rhd \mathcal{A}_3)$ is the set of ω-words $\alpha \in A_1^\omega$ such that, when $\mathcal{A}_1$ reads α, the stack of $\mathcal{A}_1$ is strictly unbounded and the limit α_1 of stack contents is in $L(\mathcal{A}_3, col_3)$. So we see that, in order to get the complement of $L(\mathcal{A}_1 \rhd \mathcal{A}_2)$ we have to add to $L(\mathcal{A}_1 \rhd \mathcal{A}_3)$ the set B of all ω-words $\alpha \in A_1^\omega$ such that, when $\mathcal{A}_1$ reads α, the stack of $\mathcal{A}_1$ is *not strictly unbounded*.
To do this we are going first to modify the automaton $\mathcal{A}_1$ in such a way that, when reading ω-words in B, the stack will be *strictly unbounded*.

We now explain informally the behaviour of the new pushdown automaton $\mathcal{A}'_1$. The stack alphabet of $\mathcal{A}'_1$ is $\Gamma_1 \cup \Gamma'_1$, where $\Gamma'_1 = \{\gamma' \mid \gamma \in \Gamma_1\}$ is just a copy of Γ_1, such that $\Gamma_1 \cap \Gamma'_1 = \emptyset$.
The essential idea is that $\mathcal{A}'_1$ will simulate $\mathcal{A}_1$ but it has the additional following behaviour. Using λ-transitions it pushes in the stack letters of Γ'_1, always keeping the information about the content of the stack of $\mathcal{A}_1$.
More precisely, if at some step while reading an ω-word $\alpha \in A_1^\omega$ by $\mathcal{A}_1$ the stack content is a finite word $\gamma = \gamma_1, \gamma_2, \ldots \gamma_j$, where each γ_i is a letter of Γ_1, then the corresponding stack content of $\mathcal{A}'_1$ will be in the form $\gamma_1.{\gamma_1'}^{n_1}\gamma_2.{\gamma_2'}^{n_2} \ldots \gamma_j.{\gamma_j'}^{n_j}$, where $n_1, n_2, \ldots, n_j$, are positive integers.
If when $\mathcal{A}_1$ reads α the stack is strictly unbounded and the limit of the stack contents is an ω-word α_1, then when $\mathcal{A}'_1$ reads the same word α its stack will be also strictly unbounded and the limit of the stack contents will be an ω-word α'_1. Moreover it will hold that $(\alpha'_1/\Gamma'_1) = \alpha_1$, where (α'_1/Γ'_1) is the word α'_1 from which are removed all letters in Γ'_1.
On the other hand if when $\mathcal{A}_1$ reads α the stack *is not strictly unbounded*, the limit of the stack contents being a finite word α_1, then when $\mathcal{A}'_1$ reads the same word α its stack *will be strictly unbounded* and its limit will be an ω-word α'_1 such that $(\alpha'_1/\Gamma'_1) = \alpha_1$.

Notice that the stack content of $\mathcal{A}'_1$ will always be in the form $\perp_1.(\perp'_1)^\star$ or $u.Z.(Z')^n$ for some $u \in \perp_1.(\Gamma_1 \cup \Gamma'_1)^\star$, $Z \in \Gamma_1$, Z' being the copy of Z in Γ'_1, and $n \geq 0$ being an integer.

The behaviour of the deterministic pushdown automaton $\mathcal{A}'_1$, reading an ω-word, will be the same as the behaviour of $\mathcal{A}_1$ but with the following modifications.

(a) Between any two transitions of $\mathcal{A}_1$ is added a λ-transition of $\mathcal{A}'_1$ which simply pushes in the stack, when the topmost stack letter of $\mathcal{A}'_1$ is $Z \in \Gamma_1$ or $Z' \in \Gamma'_1$, an additional letter Z'.

(b) Assume now that at some step of the reading of α by $\mathcal{A}'_1$ and $\mathcal{A}_1$, and after the execution of a λ-transition as explained in above item (a), the topmost stack letter of $\mathcal{A}'_1$ is some letter $Z' \in \Gamma'_1$.
Recall that the stack content of $\mathcal{A}'_1$ will be in the form $\perp_1.(\perp'_1)^n$ (if $Z' = \perp'_1$) or $u.Z.(Z')^n$ for some $u \in \perp_1.(\Gamma_1 \cup \Gamma'_1)^\star$, $Z \in \Gamma_1$, Z' being the copy of Z in Γ'_1, and $n \geq 1$.
Notice that the corresponding stack content of $\mathcal{A}_1$ will be $\perp_1$ or $(u/\Gamma'_1).Z$.
Suppose now that $\mathcal{A}_1$ reads a letter $a \in A_1$ or executes a λ-transition.
If it pushes letter T in the stack then $\mathcal{A}'_1$ would push the same letter T in its stack.
If $\mathcal{A}_1$ would skip (its topmost stack letter being Z), then $\mathcal{A}'_1$ also skips.
But if $\mathcal{A}_1$, reading the letter $a \in A_1$ or executing a λ-transition, the topmost stack letter being Z, would pop the letter Z, then $\mathcal{A}'_1$ pops the whole segment $Z.(Z')^n$ at the top of the stack, using λ-transitions.

Notice that we do not detail here the set of states of $\mathcal{A}'_1$. It contains the set of states Q_1 of $\mathcal{A}_1$ and is sufficiently enriched, to achieve the goal of simulating the behaviour of $\mathcal{A}_1$, adding the modifications cited above.

Assume now that when $\mathcal{A}_1$ reads α its stack is strictly unbounded and the limit of the stack contents is an ω-word α_1. Then when $\mathcal{A}'_1$ reads the same word α its stack is also strictly unbounded and the limit of the stack contents will be an ω-word α'_1 such that $(\alpha'_1/\Gamma'_1) = \alpha_1$.
On the other hand if when $\mathcal{A}_1$ reads α the stack *is not strictly unbounded*, then the limit of its stack contents is a finite word $\alpha_1 = \alpha_1(1).\alpha_1(2)\ldots\alpha_1(|\alpha_1|)$.
In that case when $\mathcal{A}'_1$ reads the same word α its stack *will be strictly unbounded* and its limit will be an ω-word α'_1 in the form

$$\alpha'_1 = \alpha_1(1).(\alpha_1(1)')^{n_1}.\alpha_1(2).(\alpha_1(2)')^{n_2}\ldots(\alpha_1(|\alpha_1|-1)')^{n_{|\alpha_1|-1}}.(\alpha_1(|\alpha_1|).(\alpha_1(|\alpha_1|)')^\omega$$

for some integers $n_1, n_2, \ldots, n_{|\alpha_1|-1}$. In particular it will hold that $(\alpha'_1/\Gamma'_1) = \alpha_1$.

It is now easy to modify the pushdown automaton $\mathcal{A}_3$ in such a way that we obtain a deterministic pushdown automaton $\mathcal{A}'_3$ equipped with parity acceptance condition col'_3 such that the input alphabet of $\mathcal{A}'_3$ is $\Gamma_1 \cup \Gamma'_1$, and an ω-word $\alpha'_1 \in (\Gamma_1 \cup \Gamma'_1)^\omega$ is in $L(\mathcal{A}'_3, col'_3)$ iff [(α'_1/Γ'_1) is a finite word or (α'_1/Γ'_1) is infinite and is in $L(\mathcal{A}_3, col_3)$].
Thus the ω-language $L(\mathcal{A}'_1 \rhd \mathcal{A}'_3)$ is the complement of $L(\mathcal{A}_1 \rhd \mathcal{A}_2)$ and this ends the proof. □

Proposition 3.6. For each integer $n \geq 0$, the class $\mathbb{C}_n^\lambda(A)$ is closed under complementation.

Proof. We now reason by induction on the integer $n \geq 0$.

For $n = 0$, $\mathbb{C}_0^\lambda(A) = DCFL_\omega$ is known to be closed under complementation [21].

For $n = 1$, $\mathbb{C}_1^\lambda(A)$ is closed under complementation by Lemma 3.5.

Assume now that we have proved that for every positive integer $k \leq n$ the class $\mathbb{C}_k^\lambda(A)$ is closed under complementation.

Let $\mathcal{A}_1, \mathcal{A}_2, \ldots \mathcal{A}_n, \mathcal{A}_{n+1}$, be some deterministic pushdown automata and $(\mathcal{A}_{n+2}, col)$ be a deterministic pushdown automaton equipped with a parity acceptance condition such that the language $L(\mathcal{A}_1 \triangleright \ldots \triangleright \mathcal{A}_{n+1} \triangleright \mathcal{A}_{n+2}) \subseteq A_1^\omega$ is well defined.
An ω-word $\alpha \in A_1^\omega$ is in the complement of $L(\mathcal{A}_1 \triangleright \ldots \triangleright \mathcal{A}_{n+1} \triangleright \mathcal{A}_{n+2})$ iff one the two following conditions holds:

(1) When $\mathcal{A}_1$ reads α, the stack of $\mathcal{A}_1$ is strictly unbounded and the limit α_1 of stack contents is in the complement of $L(\mathcal{A}_2 \triangleright \ldots \triangleright \mathcal{A}_{n+1} \triangleright \mathcal{A}_{n+2})$

(2) When $\mathcal{A}_1$ reads α, the stack of $\mathcal{A}_1$ is *not strictly unbounded*.

By induction hypothesis the complement of the ω-language $L(\mathcal{A}_2 \triangleright \ldots \triangleright \mathcal{A}_{n+1} \triangleright \mathcal{A}_{n+2})$ is in $\mathbb{C}_n^\lambda(A)$ so it is in the form $L(\mathcal{A}'_2 \triangleright \ldots \triangleright \mathcal{A}'_{n+1} \triangleright \mathcal{A}'_{n+2})$.
We can do similar modifications as in the case $n = 1$, replacing $\mathcal{A}_1$, whose stack alphabet is Γ_1, by another deterministic pushdown automaton $\mathcal{A}'_1$, whose alphabet is $\Gamma_1 \cup \Gamma'_1$ where Γ'_1 is a copy of Γ_1.
If when $\mathcal{A}_1$ reads α the limit of its stack contents is a finite or infinite word α_1 then when $\mathcal{A}'_1$ reads the same word α the limit of its stack contents is an ω-word α'_1 such that $(\alpha'_1/\Gamma'_1) = \alpha_1$.
It is now easy to modify the language $L(\mathcal{A}'_2 \triangleright \ldots \triangleright \mathcal{A}'_{n+1} \triangleright \mathcal{A}'_{n+2})$ in such a way that we get a language $L(\mathcal{A}''_2 \triangleright \ldots \triangleright \mathcal{A}''_{n+1} \triangleright \mathcal{A}''_{n+2})$ of ω-words over $\Gamma_1 \cup \Gamma'_1$ containing an ω-word α'_1 if and only if: either (α'_1/Γ'_1) is a finite word or (α'_1/Γ'_1) belongs to the ω-language $L(\mathcal{A}'_2 \triangleright \ldots \triangleright \mathcal{A}'_{n+1} \triangleright \mathcal{A}'_{n+2})$.

Thus it holds that $L(\mathcal{A}'_1 \triangleright \mathcal{A}''_2 \triangleright \ldots \triangleright \mathcal{A}''_{n+1} \triangleright \mathcal{A}''_{n+2})$ is the complement of $L(\mathcal{A}_1 \triangleright \ldots \triangleright \mathcal{A}_{n+1} \triangleright \mathcal{A}_{n+2})$.
$\square$

Remark 3.7. In [19, 20] Serre defined winning conditions $\Omega_{\mathcal{A}_1 \triangleright \ldots \triangleright \mathcal{A}_n \triangleright \mathcal{A}_{n+1}}$ for pushdown games using languages in classes $\mathbb{C}_n(A)$. He then showed that these winning conditions lead to decision procedures to decide the winner in pushdown games. The question now naturally arises whether the proofs can be extended to winning conditions defined in the same way from classes $\mathbb{C}_n^\lambda(A)$. Then the closure under complementation of these classes would be relevant from a game point of view. On the other hand this closure property provides also some more information about classes $\mathbb{C}_n(A)$, given by next corollary, which is already important from a game point of view.

Corollary 3.8. For each integer $n \geq 0$, the following inclusions hold:

$$\mathbb{C}_n(A) \subseteq \mathbb{C}_n^\lambda(A) \subseteq NA - CFL_\omega \bigcap Co - NA - CFL_\omega$$

Proof. It follows directly from Corollary 3.4 and Proposition 3.6. □

We now prove that the classes $\mathbb{C}_n(A)$, $\mathbb{C}_n^\lambda(A)$, are not closed under other boolean operations.

Proposition 3.9. For each integer $n \geq 0$, the classes $\mathbb{C}_n(A)$ and $\mathbb{C}_n^\lambda(A)$ are neither closed under union nor under intersection.

Proof. Notice first that for each integer $n \geq 0$, $\mathbb{C}_n(A) \subseteq \mathbb{C}_{n+1}(A)$ and $\mathbb{C}_n^\lambda(A) \subseteq \mathbb{C}_{n+1}^\lambda(A)$.

The ω-languages $L_1 = \{a^n.b^m.c^p.d^\omega \mid n,m,p \geq 1 \text{ and } n = m\}$ and $L_2 = \{a^n.b^m.c^p.d^\omega \mid n,m,p \geq 1 \text{ and } m = p\}$, over the alphabet $A = \{a,b,c,d\}$, are in $DCFL_\omega$ and they are in all classes $\mathbb{C}_n(A)$ and $\mathbb{C}_n^\lambda(A)$. But their intersection is $L_1 \cap L_2 = \{a^n.b^n.c^n.d^\omega \mid n \geq 1\}$. This ω-language is not context free because the finitary language $\{a^n.b^n.c^n \mid n \geq 1\}$ is not context free [1] and an ω-language in the form $L.d^\omega$, with $L \subseteq \{a,b,c\}^\star$, is context free iff the finitary language L is context free [7]. Thus $L_1 \cap L_2$ cannot be in any class $\mathbb{C}_n(A)$ and $\mathbb{C}_n^\lambda(A)$ because these classes are included in CFL_ω.

On the other hand consider the ω-languages $L_3 = \{a^n.b^m.c^p.d^\omega \mid n,m,p \geq 1 \text{ and } n \neq m\}$ and $L_4 = \{a^n.b^m.c^p.d^\omega \mid n,m,p \geq 1 \text{ and } m \neq p\}$. These ω-languages are in $DCFL_\omega$ and in every class $\mathbb{C}_n(A)$ or $\mathbb{C}_n^\lambda(A)$. If the language $L_3 \cup L_4$ was in some class $\mathbb{C}_n(A)$ or $\mathbb{C}_n^\lambda(A)$, then by Proposition 3.6 its complement L_5 would be also in $\mathbb{C}_n^\lambda(A)$ and it would be a context free ω-language. This would imply that $L_5 \cap a^+.b^+.c^+.d^\omega$ is context free because the class CFL_ω is closed under intersection with regular ω-languages. But $L_5 \cap a^+.b^+.c^+.d^\omega = \{a^n.b^n.c^n.d^\omega \mid n \geq 1\}$ is not context free thus for each integer $n \geq 0$, the classes $\mathbb{C}_n(A)$, $\mathbb{C}_n^\lambda(A)$ are not closed under union.

Notice that the union $\cup_{n \geq 0} \mathbb{C}_n^\lambda(A)$ is also neither closed under intersection nor under union. □

4. Winning sets in a pushdown game

Recall that it is proved in [19] that every deterministic context free language may occur as a winning set for Eve in a pushdown game equipped with a winning condition in the form $\Omega_{\mathcal{B}}$, where $\mathcal{B}$ is a deterministic pushdown automaton.
Serre asked also whether there exists a pushdown game equipped with a winning condition in the form $\Omega_{\mathcal{A}_1 \triangleright \ldots \triangleright \mathcal{A}_n \triangleright \mathcal{A}_{n+1}}$ such that the set of winning positions for Eve is not a deterministic context free language.
We are going to prove in this section that such pushdown games exist, giving examples of winning sets which are non-deterministic non-ambiguous context free languages, or inherently ambiguous context free languages, or even non context free languages.
The exact form of the winning sets remains open. Serre conjectured in [18] that one could prove that, for $n \geq 0$, the winning sets for Eve in pushdown games equipped with a winning condition in the form $\Omega_{\mathcal{A}_1 \triangleright \ldots \triangleright \mathcal{A}_n \triangleright \mathcal{A}_{n+1}}$, form a class of languages at level n, and that for $n = 0$ the winning sets could be deterministic context free languages.
So we think that, in order to better understand what is the exact form of the winning sets, it is useful to see different examples of winning sets of different complexities, and not only of the greatest complexity we have got, i.e. a non context free language.

Moreover the techniques, involving Duparc's eraser operator, used to prove Proposition 4.3 below, are interesting by their own and are useful to understand how the games go on.

In order to present the first example we begin by recalling the operation $x \to x^{\twoheadleftarrow}$ which has been introduced by Duparc in his study of the Wadge hierarchy [9], where it works also on infinite words, and is also considered by Serre in [19].

For a finite word $u \in (\Sigma \cup \{\twoheadleftarrow\})^\star$, where Σ is a finite alphabet, the finite word $u^{\twoheadleftarrow}$ is inductively defined by:

$\lambda^{\twoheadleftarrow} = \lambda$,
and for a finite word $u \in (\Sigma \cup \{\twoheadleftarrow\})^\star$:
$(u.c)^{\twoheadleftarrow} = u^{\twoheadleftarrow}.c$, if $c \in \Sigma$,
$(u.\twoheadleftarrow)^{\twoheadleftarrow} = u^{\twoheadleftarrow}$ with its last letter removed if $|u^{\twoheadleftarrow}| > 0$,
i.e. $(u.\twoheadleftarrow)^{\twoheadleftarrow} = u^{\twoheadleftarrow}(1).u^{\twoheadleftarrow}(2)\ldots u^{\twoheadleftarrow}(|u^{\twoheadleftarrow}| - 1)$ if $|u^{\twoheadleftarrow}| > 0$,
$(u.\twoheadleftarrow)^{\twoheadleftarrow} = \lambda$ if $|u^{\twoheadleftarrow}| = 0$,

Notice that for $x \in (\Sigma \cup \{\twoheadleftarrow\})^\star$, $x^{\twoheadleftarrow}$ denotes the string x, once every $\twoheadleftarrow$ occuring in x, used as an eraser, has been "evaluated" to the back space operation, proceeding from left to right inside x. In other words $x^{\twoheadleftarrow} = x$ from which every interval of the form "$c \twoheadleftarrow$ " ($c \in \Sigma$) is removed.
For a language $V \subseteq \Sigma^\star$ we set $V^\sim = \{x \in (\Sigma \cup \{\twoheadleftarrow\})^\star \mid x^{\twoheadleftarrow} \in V\}$.

Lemma 4.1. Let $L = \{a^n.b^n \mid n \geq 1\}$. Then $L^\sim$ is a non ambiguous context free language which can not be accepted by any *deterministic pushdown automaton*.

Proof. Let L be the context free language $\{a^n.b^n \mid n \geq 1\}$. The language L is a deterministic, hence non ambiguous, context free language. Thus by Theorem 6.16 of [10] the language $L^\sim$ is a non ambiguous context free language.
It remains to show that $L^\sim$ can not be accepted by any *deterministic pushdown automaton*.

The idea of the proof is essentially the same as in the proof that the context free language $\{a^n.b^n \mid n \geq 1\} \cup \{a^n.b^{2n} \mid n \geq 1\}$ can not be accepted by any *deterministic pushdown automaton*. It can be found in [1, Proof of Proposition 5.3] or in [12, Exercise 6.4.4 page 251].
Towards a contradiction assume that the language $L^\sim$ is accepted by a deterministic pushdown automaton $\mathcal{A}$. All words $a^n.b^n$, for $n \geq 1$, are in the language $L^\sim$. Then one could show that there exists a pair (n, k), with $n, k > 0$, such that the accepting configurations of $\mathcal{A}$ reading $a^n.b^n$ or $a^{n+k}.b^{n+k}$ are the same. Consider now the word $a^n.b^n.\twoheadleftarrow^{2n}.a.b$. It belongs to $L^\sim$ and the valid computation of $\mathcal{A}$ reading $a^n.b^n$ should be the beginning of the valid computation of $\mathcal{A}$ reading $a^n.b^n.\twoheadleftarrow^{2n}.a.b$. Thus the pushdown automaton $\mathcal{A}$ would also accepts $a^{n+k}.b^{n+k}.\twoheadleftarrow^{2n}.a.b$ which is clearly not in $L^\sim$. □

Lemma 4.2. Let $L \subseteq \Sigma^\star$ be a deterministic context free language. Then there exists a pushdown process $\mathcal{P} = (Q, \Gamma, \perp, \delta)$, a partition $Q = Q_E \cup Q_A$, two deterministic pushdown automata $\mathcal{A}_1, \mathcal{A}_2$, and a state $q \in Q$ such that, in the induced pushdown game equipped with the winning condition $\Omega_{\mathcal{A}_1 \triangleright \mathcal{A}_2}$, one has $\{u \mid (q, u) \in W_E\} = L^\sim$.

Proof. Let $\mathcal{P} = (\{p, q\}, \Gamma = \Sigma \cup \{\perp, \twoheadleftarrow, \#\}, \perp, \delta)$ be a pushdown process where δ is defined by: $push(p, \#) \in \delta(q, c)$ for all letters $c \in \Sigma \cup \{\perp, \twoheadleftarrow\}$ and $push(p, \#) \in \delta(p, \#)$.
So the pushdown process $\mathcal{P}$ is deterministic and its behaviour is very similar to the behaviour of the pushdown process given in the proof of Proposition 42 of [20]. It can only push the letter $\#$ on the top of a given configuration.
$Q = Q_E \cup Q_A$ is any partition of Q.
For each configuration $(q, u.c)$, for $c \in \Sigma \cup \{\perp, \twoheadleftarrow\}$ and $u \in \Gamma^\star$, there is a unique infinite play starting from $(q, u.c)$, during which the pushdown stack of $\mathcal{P}$ is strictly unbounded, and the limit of the stack contents is $u.c.\#^\omega$.

The deterministic pushdown automaton $\mathcal{A}_1$ reads words over the alphabet $\Gamma = \Sigma \cup \{\perp, \twoheadleftarrow, \#\}$ and its stack alphabet is $\Gamma_1 = \Sigma \cup \{\perp_1\}$. Its behaviour is described as follows:

Consider first the reading of an ω-word in the form $\perp.u.\#^\omega$, where $u \in (\Sigma \cup \{\twoheadleftarrow\})^\star$.
After having read the bottom symbol $\perp$, the content of its stack is still $\perp_1$. Then when the pushdown automaton $\mathcal{A}_1$ reads a letter $c \in \Sigma$ it pushes the same letter in the stack. But if $\mathcal{A}_1$ reads the symbol $\twoheadleftarrow$ and the topmost stack symbol is not $\perp_1$ (so it is in Σ) then it pops the letter at the top of its stack.

So, after having read the initial segment $\perp.u$ of $\perp.u.\#^\omega$, the stack content of $\mathcal{A}_1$ is $\perp_1.u^{\twoheadleftarrow}$. Next the PDA $\mathcal{A}_1$ pushes a letter $\#$ in the stack for each letter $\#$ read. Thus, when $\mathcal{A}_1$ reads the ω-word $\perp.u.\#^\omega$, its stack is strictly unbounded and the limit of the stack contents is $\perp_1.u^{\twoheadleftarrow}.\#^\omega$.

In addition, it is easy to ensure that, when $\mathcal{A}_1$ reads an ω-word which is not in $\perp.(\Sigma \cup \{\twoheadleftarrow\})^\star.\#^\omega \cup \perp.(\Sigma \cup \{\twoheadleftarrow\})^\omega$, then its stack *is not strictly unbounded*. If there is a letter $\perp$ after the first letter of the word or if $\mathcal{A}_1$ reads a letter in $\Sigma \cup \{\twoheadleftarrow\}$ after some letter $\#$, then the stack content remains undefinitely unchanged.

On the other hand, $\mathcal{A}_2$ is a deterministic pushdown automaton equipped with a parity acceptance condition which accepts the ω-language $\perp_1.L.\#^\omega$.
Consider now a given configuration $(q, \perp.u)$ of the pushdown process $\mathcal{P}$ for some $u \in (\Sigma \cup \{\twoheadleftarrow, \#\})^\star$, the last letter of u being not $\#$. There is a unique infinite play starting from this position. The stack of $\mathcal{P}$ is strictly unbounded during this play and the limit of stack contents is $\perp.u.\#^\omega$.
When $\mathcal{A}_1$ reads the ω-word $\perp.u.\#^\omega$ its stack is strictly unbounded iff $u \in (\Sigma \cup \{\twoheadleftarrow\})^\star$ and then the limit of stack contents is $\perp_1.u^{\twoheadleftarrow}.\#^\omega$.
The ω-word $\perp_1.u^{\twoheadleftarrow}.\#^\omega$ is accepted by $\mathcal{A}_2$ iff $u^{\twoheadleftarrow} \in L$.
Thus the configuration $(q, \perp.u)$ is a winning position for Eve in the induced pushdown game, equipped with the winning condition $\Omega_{\mathcal{A}_1 \rhd \mathcal{A}_2}$, if and only if $u \in L^{\sim}$. □

We can now state the following result which follows directly from Lemmas 4.1 and 4.2.

Proposition 4.3. There exists a pushdown process $\mathcal{P} = (Q, \Gamma, \perp, \delta)$, a partition $Q = Q_E \cup Q_A$, two deterministic pushdown automata $\mathcal{A}_1, \mathcal{A}_2$, and a state $q \in Q$ such that, in the induced pushdown game equipped with the winning condition $\Omega_{\mathcal{A}_1 \rhd \mathcal{A}_2}$, the set $\{u \mid (q, u) \in W_E\}$ is a *non-deterministic* non ambiguous context free language.

Proof. Let L be the language $\{a^n.b^n \mid n \geq 1\}$. The language L is a deterministic context free language, thus by Lemma 4.2 there exists a pushdown process $\mathcal{P} = (Q, \Gamma, \perp, \delta)$, a partition $Q = Q_E \cup Q_A$, two deterministic pushdown automata $\mathcal{A}_1, \mathcal{A}_2$, and a state $q \in Q$ such that, in the induced pushdown game equipped with the winning condition $\Omega_{\mathcal{A}_1 \rhd \mathcal{A}_2}$, one has $\{u \mid (q, u) \in W_E\} = L^\sim$. But by Lemma 4.1 $L^\sim$ is a non ambiguous context free language which can not be accepted by any *deterministic pushdown automaton*. □

Remark 4.4. In the pushdown game given in the proof of Lemma 4.2, there are some plays which are not infinite. However it is easy to find a pushdown game with the same winning set for Eve but in which all plays are infinite. The same remark will hold for pushdown games given in the proofs of the two following propositions.

We are now going to show that the set of winning positions for Eve can also be an inherently ambiguous context free language. Recall that it is well known that the language $V = \{a^n.b^m.c^p \mid n, m, p \geq 1 \text{ and } (n = m \text{ or } m = p)\}$ is an inherently ambiguous context free language, [1, 12].

Proposition 4.5. There exists a pushdown process $\mathcal{P} = (Q, \Gamma, \perp, \delta)$, a partition $Q = Q_E \cup Q_A$, two deterministic pushdown automata $\mathcal{A}_1, \mathcal{A}_2$, and a state $q \in Q$ such that, in the induced pushdown game equipped with the winning condition $\Omega_{\mathcal{A}_1 \rhd \mathcal{A}_2}$, the set $\{u \mid (q, u) \in W_E\}$ is an inherently ambiguous context free language.

Proof. Let $\mathcal{P} = (\{q, q', q'', p\}, \Gamma = \{\perp, a, b, c, \#\}, \perp, \delta)$ be a pushdown process where δ is defined by: $\{pop(q'), skip(q'')\} \subseteq \delta(q, c)$, $pop(q') \in \delta(q', c)$, $push(p, \#) \in \delta(q', b)$, $push(p, \#) \in \delta(q'', c)$, and $push(p, \#) \in \delta(p, \#)$.
We set $Q_E = \{q\}$ and $Q_A = \{q', q'', p\}$.
Consider now an infinite play from a given configuration $(q, \perp.u)$, for $u \in \{a, b, c, \#\}^\star$. The topmost stack letter of this initial configuration must be a letter c. Then at most two cases may happen.

1. In the first one are pushed infinitely many letters $\#$ on the top of the stack. In this play the stack is strictly unbounded and the limit of the stack contents is $\perp.u.\#^\omega$.

2. In the second case the letter c is popped and all next letters c are popped from the top of the stack until some letter b is on the top of the stack. From this moment infinitely many letters $\#$ are pushed in the stack. Then the stack is strictly unbounded and the limit of the stack contents is $\perp.u'.b.\#^\omega$ if $u = u'.b.c^k$ for some integer $k > 0$. Notice that this second case can only occur if u is in the form $u = u'.b.c^k$ for some integer $k > 0$.

The deterministic pushdown automaton $\mathcal{A}_1$ reads words over the alphabet $\{\perp, a, b, c, \#\}$ and its stack alphabet is $\Gamma_1 = \{\perp_1, a, b, \#\}$.
It is easy to ensure that the stack of $\mathcal{A}_1$ is not strictly unbounded during the reading of an ω-word which is not in $W = \perp.a^\omega \cup \perp.a^+.b^\omega \cup \perp.a^+.b^+.\#^\omega \cup \perp.a^+.b^+.c^\omega \cup \perp.a^+.b^+.c^+.\#^\omega$.
Consider now the reading by $\mathcal{A}_1$ of an ω-word which is in W. After having read the bottom symbol $\perp$, the stack content of $\mathcal{A}_1$ is still $\perp_1$. Then it pushes a letter a or b each time it reads the corresponding letter a or b.
Then when $\mathcal{A}_1$ reads an ω-word in the form $\perp.a^\omega$ (respectively, $\perp.a^n.b^\omega$ for $n \geq 1$) then its stack is strictly unbounded and the limit of stack contents is $\perp_1.a^\omega$ (respectively, $\perp_1.a^n.b^\omega$).

If now $\mathcal{A}_1$ reads letters $\#$ then it pushes them in the stack. In this case the input word is in the form $\perp.a^n.b^m.\#^\omega$, and the limit of stack contents of $\mathcal{A}_1$ reading this ω-word is $\perp_1.a^n.b^m.\#^\omega$.
If $\mathcal{A}_1$ reads some letters c after an initial segment in the form $\perp.a^n.b^m$ then it pops a letter b for each letter c read.
If the number of c is equal to the number of b of the input word, then after having read the segment $\perp.a^n.b^m.c^m$ of the input word the stack content of $\mathcal{A}_1$ is simply $\perp_1.a^n$. Next $\mathcal{A}_1$ reads the final segment $\#^\omega$ and it pushes it in the stack. So the limit of stack contents of $\mathcal{A}_1$ reading the input ω-word $\perp.a^n.b^m.c^m.\#^\omega$ is in the form $\perp_1.a^n.\#^\omega$
If the number of c is not equal to the number of b of the input word (the number of c being finite or infinite), then, once this has been checked, the stack content remains unchanged so the stack will not be strictly unbounded.

One can define a deterministic pushdown automaton $\mathcal{A}_2$, equipped with a parity acceptance condition, which accepts the ω-language $\{\perp_1.a^n.b^n.\#^\omega \mid n \geq 1\} \cup \{\perp_1.a^n.\#^\omega \mid n \geq 1\}$.

We are now going to determine the winning positions $(q, \perp.u)$ of Eve in the induced pushdown game equipped with the winning condition $\Omega_{\mathcal{A}_1 \rhd \mathcal{A}_2}$.

Let $(q, \perp.u)$ be a given configuration of the pushdown process $\mathcal{P}$ for some $u \in \{a, b, c, \#\}^\star$, the last letter of u being c. There are one or two infinite plays starting from this position. When there are two such plays, they depend on the first choice of Eve and the position $(q, \perp.u)$ is a winning position for Eve iff one of the two possible infinite plays is winning for her.
In the first play the stack is strictly unbounded and the limit of the stack contents is $\perp.u.\#^\omega$.
There is a second play if $u = u'.b.c^k$ for some integer $k > 0$. Then in this play the stack is strictly unbounded and the limit of the stack contents is $\perp.u'.b.\#^\omega$.

When $\mathcal{A}_1$ reads the ω-word $\perp.u.\#^\omega$, its stack is strictly unbounded iff u is in the form $a^n.b^m.c^m$ for some $n, m \geq 1$ (the number of c and of b in u are equal). Then the limit of stack contents is $\perp_1.a^n.\#^\omega$ and it is in $L(\mathcal{A}_2)$. So $\perp.u.\#^\omega \in L(\mathcal{A}_1 \rhd \mathcal{A}_2)$.

If $u = u'.b.c^k$ for some integer $k > 0$ and $\mathcal{A}_1$ reads the ω-word $\perp.u'.b.\#^\omega$ then the stack of $\mathcal{A}_1$ is strictly unbounded iff u' is in the form $a^n.b^{m-1}$ for some $n, m \geq 1$. In this case the limit of stack contents is $\perp_1.a^n.b^m.\#^\omega$ and it is accepted by $\mathcal{A}_2$ iff $n = m \geq 1$.

Thus the configuration $(q, \perp.u)$ is a winning position for Eve, with the winning condition $\Omega_{\mathcal{A}_1 \rhd \mathcal{A}_2}$, if and only if u is in the inherently ambiguous context free language $V = \{a^n.b^m.c^p \mid n, m, p \geq 1 \text{ and } (n = m \text{ or } m = p)\}$. $\square$

Proposition 4.6. There exists a pushdown process $\mathcal{P} = (Q, \Gamma, \perp, \delta)$, a partition $Q = Q_E \cup Q_A$, two deterministic pushdown automata $\mathcal{A}_1, \mathcal{A}_2$, and a state $q \in Q$ such that, in the induced pushdown game equipped with the winning condition $\Omega_{\mathcal{A}_1 \rhd \mathcal{A}_2}$, the set $\{u \mid (q, u) \in W_E\}$ is a non context free language.

Proof. We define the pushdown process $\mathcal{P} = (Q, \Gamma, \perp, \delta)$ as in the proof of preceding Proposition 4.5 except that we set this time $Q_A = \{q\}$ and $Q_E = \{q', q'', p\}$. The two deterministic pushdown automata

$\mathcal{A}_1, \mathcal{A}_2$, are also defined in the same way.

Consider now a configuration in the form $(q, \perp.a^n.b^m.c^p)$ for some integers $n, m, p \geq 1$. There are two infinite plays starting from this configuration but they depend this time on the first choice of *the second player Adam*.
The position $(q, \perp.a^n.b^m.c^p)$ is winning for Eve iff these *two infinite plays* are won by her. This implies that $n = m$ **and** $m = p$.
Thus it holds that

$$\{u \mid (q, u) \in W_E\} \cap \perp.a^+.b^+.c^+ = \perp.\{a^n.b^n.c^n \mid n \geq 1\}$$

This language is not context free because of the well known non context freeness of the language $\{a^n.b^n.c^n \mid n \geq 1\}$ [1, 12].
This implies that the set $\{u \mid (q, u) \in W_E\}$ itself is not context free. Indeed otherwise its intersection with the rational language $\perp.a^+.b^+.c^+$ would be context free because the class CFL is closed under intersection with rational languages. □

Acknowledgements. Thanks to the anonymous referee for useful comments on a preliminary version of this paper.

References

[1] Autebert, J.-M., Berstel, J., Boasson, L.: Context free languages and pushdown automata, in: *Handbook of formal languages, Vol. 1*, Springer-Verlag, 1996.

[2] Berstel, J.: *Transductions and context free languages*, Teubner Studienbücher Informatik, 1979.

[3] Bouquet, A., Serre, O., Walukiewicz, I.: Pushdown games with the unboundedness and regular conditions, in: *Proceedings of the International Conference FST TCS 2003: Foundations of Software Technology and Theoretical Computer Science, 23rd Conference*, vol. 2914 of *Lecture Notes in Computer Science*, Springer, 2003, 88–99.

[4] Cachat, T.: Symbolic strategy synthesis for games on pushdown graphs, in: *Proceedings of the International Conference ICALP 2002*, vol. 2380 of *Lecture Notes in Computer Science*, Springer, 2002, 704–715.

[5] Cachat, T.: Two-way tree automata solving pushdown games, in: *Automata, Logics, and Infinite Games* (E. Grædel, W. Thomas, T. Wilke, Eds.), vol. 2500 of *Lecture Notes in Computer Science*, Springer, 2002, 303–317.

[6] Cachat, T., Duparc, J., Thomas, W.: Solving pushdown games with a Σ_3-winning condition, in: *Proceedings of the 11th Annual Conference of the European Association for Computer Science Logic, CSL 2002*, vol. 2471 of *Lecture Notes in Computer Science*, Springer, 2003, 322–336.

[7] Cohen, R., Gold, A.: Theory of ω-languages, parts one and two, *Journal of Computer and System Science*, **15**, 1977, 169–208.

[8] Cohen, R., Gold, A.: ω-computations on deterministic pushdown machines, *Journal of Computer and System Science*, **16**, 1978, 275–300.

[9] Duparc, J.: Wadge hierarchy and Veblen hierarchy: Part 1: Borel sets of finite rank, *Journal of Symbolic Logic*, **66**(1), 2001, 56–86.

[10] Finkel, O.: Ambiguity in omega context free languages, *Theoretical Computer Science*, **301**(1-3), 2003, 217–270.

[11] Gimbert, H.: Parity and exploration games on infinite graphs, in: *Proceedings of the 13th Annual Conference of the European Association for Computer Science Logic, CSL 2004*, vol. 3210 of *Lecture Notes in Computer Science*, Springer, 2004, 56–70.

[12] Hopcroft, J. E., Motwani, R., Ullman, J. D.: *Introduction to automata theory, languages, and computation*, Addison-Wesley Publishing Co., Reading, Mass., 2001, Addison-Wesley Series in Computer Science.

[13] Hopcroft, J. E., Ullman, J. D.: *Introduction to automata theory, languages, and computation*, Addison-Wesley Publishing Co., Reading, Mass., 1979, Addison-Wesley Series in Computer Science.

[14] Kechris, A. S.: *Classical descriptive set theory*, Springer-Verlag, New York, 1995.

[15] Kuratowski, K.: *Topology*, Academic Press, New York, 1966.

[16] Lescow, H., Thomas, W.: Logical specifications of infinite computations, in: *A Decade of Concurrency* (J. W. de Bakker, W. P. de Roever, G. Rozenberg, Eds.), vol. 803 of *Lecture Notes in Computer Science*, Springer, 1994, 583–621.

[17] Serre, O.: Note on winning positions on pushdown games with ω-regular conditions, *Information Processing Letters*, **85**, 2003, 285–291.

[18] Serre, O.: *Contribution à l'étude des jeux sur des graphes de processus à pile*, Ph.D. Thesis, Université Paris VII, 2004.

[19] Serre, O.: Games with winning conditions of high borel complexity, in: *Proceedings of the International Conference ICALP 2004*, vol. 3142 of *Lecture Notes in Computer Science*, Springer, 2004, 1150–1162.

[20] Serre, O.: Games with winning conditions of high borel complexity, 2004, To appear in *Theoretical Computer Science*, available from : http://www.liafa.jussieu.fr/ serre.

[21] Staiger, L.: ω-languages, in: *Handbook of formal languages, Vol. 3*, Springer, Berlin, 1997, 339–387.

[22] Thomas, W.: Automata on infinite objects, in: *Handbook of Theoretical Computer Science* (J. van Leeuwen, Ed.), vol. B, Formal models and semantics, Elsevier, 1990, 135–191.

[23] Thomas, W.: On the synthesis of strategies in infinite games, in: *Proceedings of the International Conference STACS 1995*, vol. 900 of *Lecture Notes in Computer Science*, Springer, 1995, 1–13.

[24] Thomas, W.: Infinite games and verification (extended abstract of a tutorial), in: *Proceedings of the International Conference on Computer Aided Verification CAV 2002*, vol. 2404 of *Lecture Notes in Computer Science*, Springer, 2002, 58–64.

[25] Walukiewicz, I.: Pushdown processes: games and model checking, *Information and Computation*, **157**, 2000, 234–263.